บุญเลิศ เอี่ยมทัศนา

SOFT MAIL

สนุกกับ เมตาไฟล์ ตอนที่ 1

นำเสนอเรื่องของการอ่านและเขียนไฟล์เพลสเอเบิลเมตาไฟล์ตลอดจนการนำออกแสดง

ในบ้านเราเมื่อซื้อเครื่องจะได้โปรแกรมสารพัดโปรแกรมจำนวนหนึ่ง กับไฟล์นานาชนิด อาจมีเมตาไฟล์ด้วย สำหรับเมตาไฟล์หากปล่อยทิ้งไว้ก็เป็นที่น่าเสียดาย บทความนี้จะนำมันมาใช้ ประกอบด้วยเรื่องอื่นอีก จึงต้องแบ่งออกเป็นสองตอน

เมตาไฟล์กับบิตแม็พ

คู่แข่งของเมตาไฟล์คือบิตแม็พ เพราะต่างก็ใช้สำหรับเก็บภาพไว้แสดงทั้งคู่ ข้อแตกต่างที่สำคัญคือ บิตแม็พเก็บภาพที่วาดด้วยฟังก์ชันวาดภาพ ส่วนเมตาไฟล์เก็บคำสั่งที่ใช้ในการวาดภาพ ทำให้เกิดข้อแตกต่างต่าง ๆ คือ

1. ไฟล์ของเมตาไฟล์จะมีขนาดเล็กกว่าบิตแม็พมาก

2. สามารถแสดงภาพซ้อนทับกัน (มองทะลุช่องว่าง) ได้ ซึ่งกับบิตแม็พทำไม่ได้

3. สามารถเลื่อนภาพเล็กในภาพใหญ่ได้ ซึ่งกับบิตแม็พทำไม่ได้เช่นกัน

4. สามารถย่อและขยายภาพโดยคงภาพเดิมได้ดีกว่าบิตแม็พ

5. การแสดงภาพแสดงได้ช้ากว่าบิตแม็พ

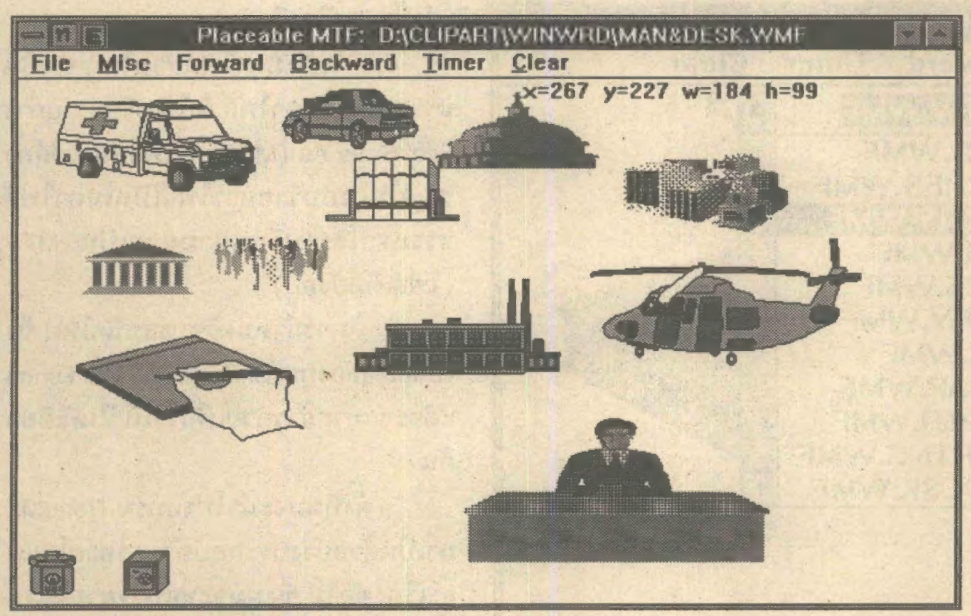
6. หากต้องการให้ใช้งานได้ทั่วไปจะต้องใช้สีเพียง 16 สี ต่างกับบิตแม็พที่อาจถือได้ว่าไม่จำกัด

โปรแกรมนี้

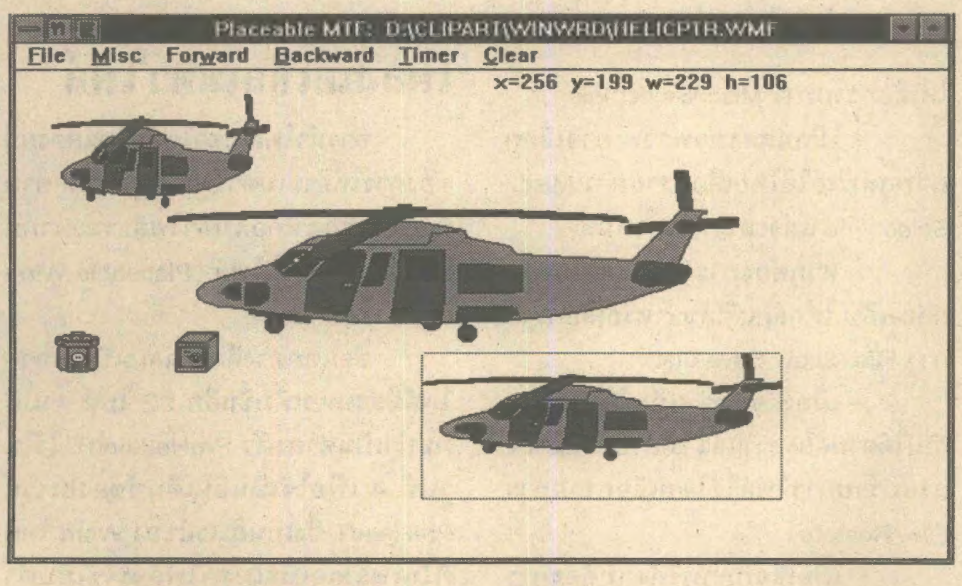
ซึ่งผมให้ชื่อว่า ReadMtf เป็นโปรแกรมอ่านเมตาไฟล์ทั้งธรรมดาและเพลสเอเบิลเมตาไฟล์ แล้วนำออกแสดงทั้งเพื่อดูหรือเพื่อการคัดเลือก ซึ่งอาจคัดเลือกไว้แสดงในครั้งต่อไป หรือเพื่อเปลี่ยนเพลสเอเบิลเมตาไฟล์เป็นเมตาไฟล์ธรรมดา เพื่อทำเป็นรีซอร์สในโปรแกรมอื่นต่อไป จึงมีเรื่องต่าง ๆ มากล่าวดังนี้

1. การอ่านไฟล์ เมื่อรันโปรแกรมนี้จะแสดงหน้าต่างดังรูปที่ 1 แต่ยังไม่มีภาพในหน้าต่าง ดังที่เห็นคงมีแต่ภาพไอคอน 2 ไอคอนตรงมุมล่างซ้าย ผู้ใช้สามารถเลือกรายการ File/Open เพื่ออ่านไฟล์มาแสดงได้ตามปกติ

2. แดร็กไฟล์เป็นกลุ่ม แต่การใช้ File/Open เพื่ออ่านไฟล์เป็นจำนวนมากครั้งละ 1 ไฟล์ไม่สนุกนักครับ จึงให้สามารถดำเนินการได้ด้วยวิธีแดร็กแอนด์ดร็อบ (drag and drop) วิธีหนึ่งคือให้เลือกไฟล์เป็นกลุ่มในหน้าต่างของ File



รูปที่ 1 หน้าต่างของ ReadMtf เมื่อนำภาพหลายภาพออกแสดง มีไอคอนของถังผงและกล่องเก็บอยู่ตรงมุมล่างซ้าย



รูปที่ 2 แสดงถึงการแสดงภาพด้วยการตีกรอบ

Manager แล้วแดร็กคือคลิ๊กเม้าส์ซ้ายในกลุ่มไฟล์ที่เลือก ไปดร็อบคือไปปล่อยมือในหน้าต่างของ ReadMtf

วิธีการนี้ไม่ค่อยนำมากล่าวถึงกันมีอยู่ในตัวอย่างของเทอร์โบปาสคาล (shelldem.pas) เพื่อแยกกับวิธีอื่นจึงขอเรียกเป็นการแดร็กไฟล์เป็นกลุ่ม หรือการแดร็กแอนด์ดร็อบไฟล์เป็นกลุ่ม

3. แดร็กไฟล์เดี่ยว เพื่อให้มีเรื่องของการแดร็กแอนด์ดร็อบครบชุด จึงนำวิธีแดร็กแอนด์ดร็อบที่รู้จักกันทั่วไปมาใช้ด้วย คือที่ให้ผู้ใช้แดร็กไฟล์ข้อมูลไปดร็อบบนไฟล์โปรแกรม เช่น .exe ซึ่งจะเป็นทั้งการรันโปรแกรมนั้น และถ้าโปรแกรมนั้นรับการดร็อบแบบนี้ก็จะนำไฟล์ข้อมูลมาใช้งาน เช่น อ่านมาแสดง และทำให้สามารถจัดเป็นไฟล์คู่ได้ด้วยรายการ File/Associate ใน File Manager

วิธนี้มีในตัวอย่างของเทอร์โบปาสคาลแทบทุกตัวอย่าง เพียงแต่ไม่บอกว่าเพื่อการใด (หรือผมหาไม่พบ)

4. การแสดงภาพ เมื่อมีไฟล์เป็นจำนวนมาก และเพื่อแสดงขีดความสามารถของเมตาไฟล์ด้วย จึงจัดให้มีการแสดงภาพได้ดังนี้คือ

เมื่อโหลดไฟล์หรือเมื่อแดร็กแอนด์ดร็อบจะแสดงภาพตามขนาดที่ผู้วาดกำหนดไว้ ดังภาพเฮลิคอปเตอร์ลำกลางในรูปที่ 2 ซึ่งสามารถที่จะ

- แสดงภาพขนาดต่าง ๆ ได้ด้วยการตีกรอบคือ คลิ๊กเม้าส์ซ้ายที่มุมบนซ้ายไปปล่อยมือที่มุมล่างขวา จะแสดงภาพในกรอบนี้ดังภาพเฮลิคอปเตอร์อีก 2 ลำ

- หากต้องการให้แสดงกรอบด้วยให้เช็ครายการ Misc/With Rectangle ดังลำล่าง

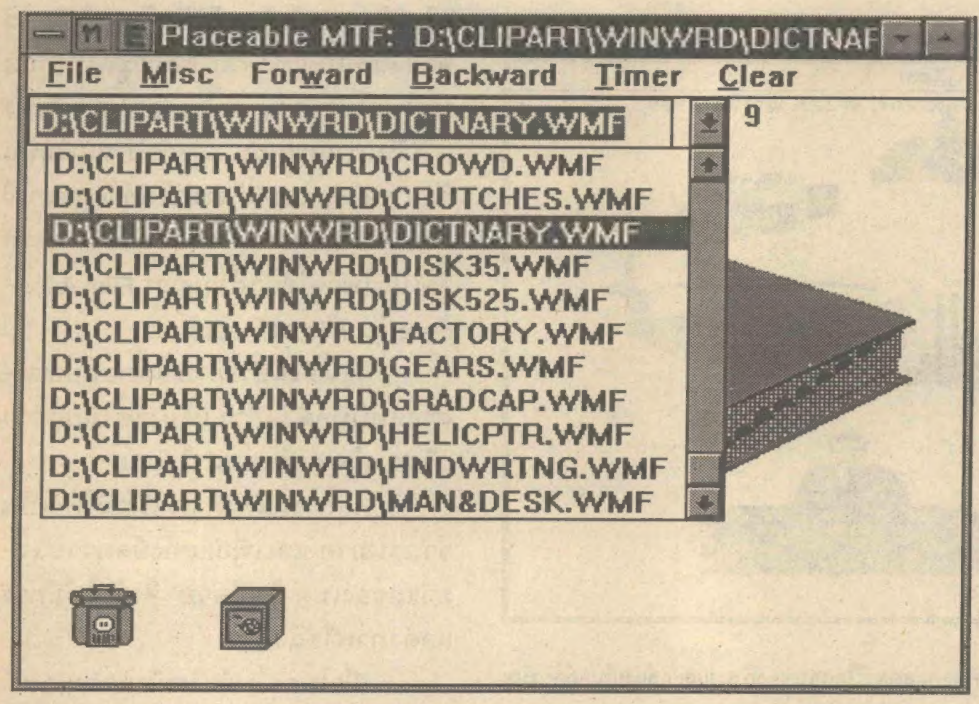
- หากต้องการรักษาสัดส่วนเดิมไว้ ให้เช็ครายการ Misc/Use Width to Scale ดังลำล่างเช่นกัน

เมื่อแดร็กไฟล์เป็นกลุ่มมาสามารถเลือกภาพออกแสดงได้ดังนี้

- เลือกรายการ Forward เพื่อให้แสดงภาพถัดไป และ Backward เพื่อย้อนกลับ

- เลือกรายการ Timer/Start-Timer เพื่อให้แสดงภาพหมุนเวียนตามช่วงเวลา

- เลือกรายการ File/Show Files List จะแสดงกรอบรายการไฟล์ดังรูปที่ 3



รูปที่ 3 รายการไฟล์ในกรอบรายการไฟล์

ให้เลือกไฟล์ที่ต้องการ จะแสดงภาพทันที

- เลือกรายการ Misc/Muttiple display แล้วตีกรอบ จะเปลี่ยนภาพทุกครั้งที่ตีกรอบ คือที่เห็นในรูปที่ 1 และจะขอเรียกเป็นการแสดงภาพรวม

5. การเลือกภาพ เพื่อให้ผู้ใช้เลือกภาพได้โดยสะดวก จึงนำวิธีแดร็กแอนด์-ดร็อบอีกวิธีหนึ่งมาใช้ กล่าวคือ

- เมื่อต้องการเลือกภาพใดให้แดร็กภาพนั้นไปดร็อบที่ถังผง (ไอคอนซ้าย) เมื่อต้องการคัดออก หรือกล่องเก็บ (ไอคอนขวา) เมื่อต้องการเก็บไว้ (ดูในรูปที่ 3 จะเห็นไอคอนชัดกว่า)

- หากไม่ได้แสดงภาพรวม ภาพเดิมจะหายไปและแสดงภาพถัดไป เพื่อให้เลือกภาพได้ต่อเนื่อง

- การเลือกภาพนี้จะทำได้ 1 ครั้งต่อ 1 ภาพ หากต้องการตีกรอบให้คลิ๊กเม้าส์ก่อน 1 ครั้ง หรือคลิ๊กเม้าส์นอกกรอบภาพ และถ้าต้องการกลับมาเลือกภาพใหม่ให้เลือกรายการ Misc/Select File

- เมื่อแสดงภาพรวม อาจเลือกภาพสุดท้ายได้โดยเลือกรายการ Misc/Select File แต่จะลบภาพทั้งหมด

- หากต้องการดูรายการไฟล์ในกล่องเก็บ ให้คลิ๊กเม้าส์ขวาหรือเลือกรายการ File/Show Save List

- เมื่อเลือกภาพเก็บไว้ในกล่องเก็บได้ตามต้องการแล้ว ให้ถ่ายกลับมายังกรอบรายการไฟล์ได้โดยเลือกรายการ File/Restore

- เมื่อเลือกภาพได้ตามต้องการแล้ว อาจเก็บรายการไฟล์ในกรอบรายการไฟล์ไว้ในเท็กซ์ไฟล์ได้โดยเลือกรายการ Misc/Save List as Text และสามารถอ่านกลับมาแสดงได้โดยเลือกรายการ Load From Text เช่น ในกรณีรันโปรแกรมนี้ครั้งต่อไป

6. ทำเป็นรีซอร์ส สามารถทำเมตาไฟล์ธรรมดาเป็นรีซอร์สได้เช่นกัน แยกได้เป็น 2 เรื่องคือ

- ไฟล์เมตาไฟล์ส่วนใหญ่จะเป็นเพลสเอเบิลเมตาไฟล์ จึงได้เตรียมรายการ File/Save As (MTF) ไว้เพื่อเก็บไฟล์ทั้งหมดในกรอบรายการไฟล์เป็นเมตาไฟล์ธรรมดาโดยเปลี่ยนนามสกุลเป็น MTF (ไฟล์เดิมยังอยู่)

- การกำหนดรีซอร์สชนิดใหม่ ซึ่งเป็นเรื่องของการใช้ Resaurce Workshop แต่จะนำมากล่าวด้วยเพื่อรวมไว้ในที่เดียวกัน

ดังที่ผมจะนำโปรแกรม ReadMtf มาเป็นตัวอย่างประกอบคำบรรยายในแต่ละเรื่องต่อไป ซึ่งผมจะขอบรรยายโปรแกรมในส่วนที่เป็นเรื่องหลัก สำหรับเรื่องปลีกย่อยผมคิดว่าอ่านได้จากโปรแกรมโดยตรง

เพลสเอเบิลเมตาไฟล์

การนำไฟล์เมตาไฟล์ออกแสดงผมต้องการให้อ่านเมตาไฟล์ได้ทุกชนิด ตามที่รู้จักในขณะนี้คือ เมตาไฟล์ธรรมดากับเพลสเอเบิลเมตาไฟล์ (Placeable Windows Metafiles)

ข้อแตกต่างคือเพลสเอเบิลเมตาไฟล์มีส่วนหัวเพิ่มขึ้นอีก 22 ไบต์ ตามที่ผมทำเป็นค่าคงตัว PreHeadIndT ไว้ในรูปที่ 4 เพื่อใช้เป็นอินเด็กซ์ของอะเรย์ PreHeadT ซึ่งเป็นอะเรย์ของ Word โดยที่ในเพลสเอเบิลเมตาไฟล์จะใช้เก็บค่าต่าง ๆ ดังนี้

- key1 และ key2 จะต้องมีค่าเป็น $CDD7 และ $9AC6 ตามลำดับ (ตามที่ทำเป็นค่าคงตัว wrd0 และ wrd1 ไว้)

- hmf ไม่ใช้ต้องให้ค่าเป็นศูนย์

- px1 กับ py1 และ px2 กับ py2 คือจุดมุมบนซ้ายและล่างขวาของกรอบสี่เหลี่ยมที่เล็กที่สุดที่ครอบภาพทั้งหมดใน

```
TYPE
  PreHeadIndT = (key1,key2,hmf,px1,py1,px2,py2,inch,rsv1,rsv2,chck);
  PreHeadT    = array[PreHeadIndT] of Word;

CONST
    wrd0  = $CDD7;  wrd1  = $9AC6;

inch    Specifies the number of metafile units to the inch.
        To avoid numeric overflow, this value should be less than 1440.
        Most applications use 576 or 1000.


 TMetaHeader = record
    mtType : Word;
    mtHeaderSize : Word;
    mtVersion : Word;
    mtSize : Longint;
    mtNoObjects : Word;
    mtMaxRecord : Longint;
    mtNoParameters : Word;
  end;
```

รูปที่ 4 ค่าในส่วนหัวเพิ่มของเพลสเอเบิลเมตาไฟล์ และค่าส่วนหัวของเมตาไฟล์

เมตาไฟล์ ก็คือกรอบภาพครับ มีหน่วยตามค่าอินช์

- inch ขอเรียกเป็นค่าอินช์ด้วย มีความหมายตามคำนิยามใน Help ของเทอร์โบปาสคาล ตามที่ผมลอกมาแสดงในรูปที่ 4 คือจำนวนจุดเป็น LU (logical unit) ต่อ 1 นิ้ว

- rsv1 และ rsv2 เป็นเวิร์ดสำรอง 2 เวิร์ด

- chck เป็นเวิร์ดตรวจสอบ เก็บค่าที่ได้จากการ XOR ค่าใน 10 เวิร์ดแรก

ถัดจากส่วนหัวเพิ่มคือส่วนหัวของเมตาไฟล์ตามปกติ ดังที่ได้ทำเป็นแบบข้อมูล TMetaHeader ไว้ดังในรูปที่ 4 มีฟิลด์ข้อมูลดังนี้ คือ

- mtType ทั้งในคู่มือวินโดวส์และใน Help ของเทอร์โบปาสคาลกล่าวไว้ตรงกันว่า ถ้ามีค่าเป็น 1 หมายถึงเมตาไฟล์ในหน่วยความจำ ถ้าเป็น 2 หมายถึงเมตาไฟล์ในดิสก์ แต่ที่เป็นจริงพบว่าเมตาไฟล์ในดิสก์มีค่าเป็น 1

- mtHeaderSize คือขนาดของเรคอร์ดนี้เป็นเวิร์ด

- mtVersion ถ้าเป็นวินโดวส์รุ่น 3.0 เป็นต้นไป มีค่าเป็น $0300 รุ่นก่อนมีค่าเป็น $0100

- mtSize ขนาดของไฟล์เป็นเวิร์ด

- mtNoObject จำนวนออปเจ็กต์สูงสุดที่ใช้อยู่ในขณะเดียวกัน (ออปเจ็กต์ได้แก่ ปากกา แปรง ฟอนต์ บิตแม็พ เขตพื้นที่ และจานสี)

- mtMaxRecord ขนาดของเรคอร์ดที่ใหญ่ที่สุดในไฟล์นั้น

- mfNoParameter เป็นเวิร์ดสำรอง

การอ่านไฟล์เมตาไฟล์

การอ่านไฟล์เพลสเอเบิลเมตาไฟล์ประกอบด้วยขั้นตอนเป็นส่วนรวมดังนี้คือ

1. อ่านไฟล์มาตัดส่วนหัวเพิ่มออก คือ อ่านส่วนหัวเพิ่มมาเก็บไว้เพื่อใช้ในการแสดงภาพ แล้วอ่านส่วนที่เหลือให้แก่หน่วยความจำ

2. แปลงค่าในหน่วยความจำให้เป็นเมตาไฟล์ภาคแสดงด้วยฟังก์ชัน SetMetaFileBits

ขั้นตอนต่อไปคือนำเมตาไฟล์ภาคแสดงไปใช้ เช่นเดียวกับที่เปิดเมตาไฟล์ธรรมดาด้วยฟังก์ชัน GetMetaFile

โปรแกรมอ่านเมตาไฟล์

แต่เมื่อนำมาเขียนโปรแกรมจะต้องประกอบด้วยขั้นตอนย่อยอื่นอีก เพื่อตรวจสอบข้อผิดพลาดดังโปรแกรมของฟังก์ชัน LoadMtf ในลิสติงที่ 1 เป็นโปรแกรมที่ให้อ่านได้ทั้งเมตาไฟล์ธรรมดาและเพลสเอเบิลเมตาไฟล์

ฟังก์ชันนี้จะอ่านไฟล์เมตาไฟล์ตามชื่อไฟล์ที่ให้แก่ fiNmeP มาทำเป็นเมตาไฟล์ภาคแสดงให้ค่าแฮนเดิลแก่ตนเองส่งกลับไป พร้อมด้วยค่าส่วนหัวเพิ่มใน preHde ค่าส่วนหัวของเมตาไฟล์ใน mtaHde และถ้าเป็นเพลสเอเบิลเมตาไฟล์จะให้ isPlceMtf เป็นจริง โดยที่ในการตรวจสอบหากพบข้อขัดข้องได้ผลเป็นเท็จจะให้ตนเองเป็นศูนย์ส่งกลับไป ตามที่ได้แบ่งโปรแกรมไว้เป็น 6 ท่อนคือ

- ท่อนแรก ตรวจว่ามีไฟล์หรือไม่ ถ้ามีไฟล์จะอ่านค่าที่ต้นไฟล์ให้แก่ temhde ซึ่งขณะนี้ยังไม่ทราบว่าเป็นส่วนหัวเพิ่มหรือส่วนหัวของเมตาไฟล์

- ท่อนที่สอง ตรวจสอบว่าเป็นเมตาไฟล์ธรรมดาหรือเป็นเพลสเอเบิลเมตาไฟล์หรือไม่

- ท่อนที่สาม หากเป็นเมตาไฟล์ธรรมดาให้ปิดไฟล์แล้วเปิดไฟล์ด้วยฟังก์ชัน GetMetaFile ได้เป็นค่าแฮนเดิลของเมตาไฟล์ภาคแสดงส่งกลับไป (โปรแกรมที่เหลือในท่อนนี้และในท่อนอื่นด้วย คือที่ให้ค่าแก่ mtaHde ผมเตรียไว้สำหรับงานอื่น)

- หากไม่ใช่เมตาไฟล์ธรรมดา จะเป็นเพลสเอเบิลเมตาไฟล์ตามที่จะดำเนิน

ลิสติงที่ 1

```
(* ldesvwmf.icl *)

(* ------------------ Load and Save Metafile ---------- *)

CONST
    wrd0  = $CDD7;  wrd1  = $9AC6;
    dwx   = 288;    dwy   = 288;
    maxIO = 65532;  oneIO = 32766;
CONST
  defiso : PreHeadT =
    (wrd0,wrd1,0,dwx,dwy,dwx+576*3,dwy+576*2,576,0,0,0);


PROCEDURE  OpenFiErr(hfi : Integer; err : Integer);
VAR  buf : array[0..127] of Char;  P : PChar;
BEGIN
  if  hfi <> 0  then  _lclose(hfi);
  lstrcpy(buf, 'Can not ');
  case  err  of
   1001 : P := 'open file.';
   1002 : P := 'read file.';
   1003 : P := 'load this meta file type.';
   1004 : P := 'load file larger then 64K byte.';
   1005 : P := 'allocate memory.';
   1006 : P := 'load file';
   1007 : P := 'set memory meta file';
   1008 : P := 'get error';
   1009 : P := 'open file for save';
  end;  { case }
  lstrcat(buf, P);
  MessageBox(GetFocus, buf, 'Error', mb_OK + mb_IconStop);
END;


FUNCTION  _lFileSize(hfi : Integer) : LongInt;
VAR  curpos : LongInt;
BEGIN
  curpos := _llseek(hfi, 0, 1);
  _lFileSize := _llseek(hfi, 0, 2);
  _llseek(hfi, curpos, 0);
END;


FUNCTION  LoadMtf(fiNmeP : PChar; var preHde : PreHeadT;
                  var mtaHde : TMetaHeader;
                  var isPlceMtf : Boolean) : THandle;
VAR  hfi : Integer;   hMem,hMtaPlay : Thandle;
     mtsi : LongInt;  mtfP : PChar;  temhde : PreHeadT;
BEGIN
  LoadMtf := 0;
  hfi := _lopen(fiNmeP, of_Read);
  if  hfi = -1  then  begin  OpenFiErr(0,1001);  Exit  end;
  mtsi := _lFileSize(hfi);
  if _lread(hfi, @temhde, SizeOf(temhde)) <> SizeOf(temhde)  then
    begin  OpenFiErr(hfi,1002);  Exit  end;

  if  (temhde[key1] <> 1) and
      ((temhde[key1] <> wrd0) or (temhde[key2] <> wrd1)) then
    begin  OpenFiErr(hfi,1003);  Exit  end;

  if  temhde[key1] = 1  then
    begin  _lclose(hfi);  LoadMtf := GetMetaFile(fiNmeP);
           Move(temhde, mtaHde, SizeOf(TMetaHeader));
           mtaHde.mtNoParameters := mtsi;
           isPlceMtf := false;   Exit  end;

  isPlceMtf := true;  mtsi := mtsi - SizeOf(temhde);
  hMem := GlobalAlloc(gmem_Moveable, mtsi);
  if  hMem = 0  then
    begin  OpenFiErr(hfi,1005);  Exit  end;
  mtfP := GlobalLock(hMem);
  if  mtfP = nil  then
    begin  GlobalFree(hMem);  OpenFiErr(hfi,1005);  Exit  end;

  if HugeIO(_LRead, hFi, mtfP, mtsi) = 0  then
    begin  GlobalUnlock(hMem);   GlobalFree(hMem);
           OpenFiErr(hfi,1006);  Exit  end;
  _lclose(hfi);
```

การในโปรแกรมท่อนต่อไป

- ท่อนที่สี่ เป็นการจองหน่วยความจำ

- ท่อนที่ห้า โหลดข้อมูลลงหน่วยความจำ

- ท่อนที่หก คือการแปลงเป็นเมตาไฟล์ภาคแสดงด้วย SetMetaFileBits กับถ่ายค่าใน temhde เป็น PreHde ส่งกลับไปเพื่อนำไปใช้ในการกำหนดสเกล

โปรแกรมเขียนกับไฟล์เมตาไฟล์

ในการเขียนกับเมตาไฟล์ คงดำเนินการตรงข้ามกัน ตามโปรแกรมของฟังก์ชัน SaveMtf ในลิสติงที่ 1 ฟังก์ชันนี้จะเขียนกับไฟล์ตามค่าที่ผ่านมาให้ดังนี้

- fiNmeP เป็นชื่อไฟล์

- hSave เป็นเมตาไฟล์ภาคแสดง

- preHde เป็นส่วนหัวเพิ่ม

และจะให้ค่าเป็นจริงเมื่อเขียนกับไฟล์ได้เป็นเท็จเมื่อเขียนไม่ได้ กล่าวคือ

- แปลงเมตาไฟล์ภาคแสดงให้เป็นค่าแฮนเดิลของหน่วยความจำด้วย GetMetaFileBits (คือตรงข้ามกับ SetMetaFileBits) แล้วแปลงค่าแฮนเดิลเป็นพอยน์เตอร์

- เปิดไฟล์

- เขียนข้อมูลของส่วนหัวเพิ่ม

- เขียนข้อมูลของเมตาไฟล์ จะเห็นว่าได้นำขนาดคือจำนวนไบต์ของข้อมูลที่จะเขียนกับไฟล์มาจากฟิลด์ mtSize ซึ่งวินโดวส์จะเป็นผู้ให้ค่านี้เป็นเวิร์ด เสร็จแล้วให้ปิดไฟล์

- แปลงค่าหน่วยความจำที่เป็นเมตาไฟล์คือ hMem กลับเป็นเมตาไฟล์ภาคแสดงให้แก่ hSave ทั้งนี้เพราะฟังก์ชัน hSave จะยกเลิกค่าใน hSave

ลิสติงที่ 1 (ต่อ)

```
    Move(mtfP^, mtaHde, SizeOf(TMetaHeader));
    mtaHde.mtNoParameters := mtsi;  GlobalUnlock(hMem);
    hMtaPlay := SetMetaFileBits(hMem);  LoadMtf := hMtaPlay;
    if  hMtaPlay = 0  then
      begin  GlobalFree(hMem);  OpenFiErr(0,1007)  end
    else  preHde := temhde;
  END;


  FUNCTION  SaveMtf(fiNmeP : PChar; var hSave : THandle;
                    var preHde : PreHeadT) : Boolean;
  VAR  hMem : THandle;  mtfP : PChar;  mtSi : LongInt;
       mtaHdeP : PMetaHeader;   hFI : Integer;
  BEGIN
    hMem := GetMetaFileBits(hSave);
    mtfP := GlobalLock(hMem);
    mtaHdeP := PMetaHeader(mtfP);
    mtSi := mtaHdeP^.mtSize*2;  SaveMtf := false;
    _lcreat(fiNmeP,0);  hFI := _lopen(fiNmeP, of_Write);
    if  hFI = -1  then
      begin  MessageBox(GetFocus, 'Can''t open file for save',
                        'Error', mb_OK + mb_IconStop);  Exit  end;

    _lwrite(hFI, @preHde, SizeOf(preHde));

    HugeIO(_lwrite, hFi, mtfP, mtSi);
    _lclose(hFI);  SaveMtf := true;
    GlobalUnlock(hMem);  SetMetaFileBits(hMem);
  END;
```

ลิสติงที่ 2

```
PROGRAM   ReadMtf;  { readmtf.pas  }
{$R readmtf.res}
{USES   WinTypes,WinProcs,WObjects,StdWnds,Strings,WinDos,
       UBkUtils,Win31,ShellAPI; }  { TPW }
USES   WinTypes,WinProcs,OWindows,OStdWnds,ODialogs,Windos,Strings,
       UBkUtils,Win31,ShellAPI;  { BPW }

CONST  appname  = 'readmtf';
       thetitle = 'Read and Drop MTF';
       menuname = 'readmtf_menu';
       editmenu = 'FileCommands';
       deliconme = 'delico_ico';
       sveiconme = 'sveico_ico';

       txthead  = 'MTF file list';

   id_CombBox = 51;  id_ListBox = 52;
   cm_NewFi  = 101;  cm_OpenFi = 102;  cm_SveMtf = 103;
   cm_ShwLst = 104;  cm_ClrLst = 105;  cm_ShwSve = 106;
   cm_ReStor = 107;  cm_Note   = 108;  cm_Quit   = 109;
   cm_SveLst = 201;  cm_LdeTxt = 202;  cm_MulDsp = 203;
   cm_SelFi  = 204;  cm_Draft  = 205;  cm_UseWid = 206;
   cm_OnTop  = 209;  cm_Fwrd   = 300;  cm_Bwrd   = 400;
   cm_SttTmr = 501;  cm_StpTmr = 502;  cm_Fast   = 503;  cm_Slow   = 504;
   cm_ClrWnd = 600;


   cmbWidK = 300;  TmrIdK    = 1;    TmeIntvK   = 3000;


TYPE
  PreHeadIndT = (key1,key2,hmf,px1,py1,px2,py2,inch,rsv1,rsv2,chck);
  PreHeadT = array[PreHeadIndT] of Word;

  TwoPntsT = record  fpt1,fpt2 : TPoint  end;
  TwoPntsArrayTP = ^TwoPntsArrayT;
  TwoPntsArrayT  = array[0..10] of TwoPntsT;

{$I ldesvwmf.icl}
{ $I icownd.icl}

TYPE
  FiNameT = array[0..144] of Char;

  WndProCP = ^WndProC;
  WndProC  = OBJECT(TWindow)
    mtfFNB : FiNameT;
    oFiListP : PComboBox;
```

เดิม แต่โปรแกรมที่เรียกมาอาจต้องการใช้และควรเป็นผู้ยกเลิกค่านี้ด้วยตนเอง

ไฟล์ ldesvmtf.icl

โปรแกรมอ่านเขียนกับเพลส-เอเบิลเมตาไฟล์ตามลิสติงที่ 1 ผมเก็บไว้ในไฟล์ ldesvmtf.icl เพื่อที่จะผนวก (include) เข้ากับไฟล์ readmtf.pas ซึ่งเป็นไฟล์หลักของโปรแกรมนี้

ทั้งนี้เพื่อเตรียมใช้กับโปรแกรม WmfPaint ตามที่จะนำมากล่าวเป็นเรื่องต่อไปด้วย ซึ่งในการนำมาใช้จะต้องจัดกิจกรรมเรียกใช้โปรแกรมย่อยทั้งสองนี้อีกทอดหนึ่ง

โปรแกรม ReadMtf

โปรแกรม ReadMtf อยู่ในลิสติงที่ 2 เป็นโปรแกรมที่จะต้องต่อเติมในตอนหน้า และได้เตรียมไว้ให้ต่อเติมได้ง่ายด้วย คือ

1. ได้เตรียมการใช้สำหรับผนวกไฟล์ icownd.icl แต่เนื่องจากยังไม่มีไฟล์นี้ จึงแทรกวรรคไว้ข้างหน้าเครื่องหมาย $

2. ในการกำหนดสกุล WndProC มีกิจกรรมส่วนหนึ่ง (อยู่ตอนท้าย) ยังไม่มีโปรแกรม จึงได้ทำเป็นหมายเหตุไว้

3. และเมื่อเรียกใช้กิจกรรมเหล่านี้ใน WmCommand ได้ทำเป็นหมายเหตุไว้เช่นกัน

คงเหลือโปรแกรมของฉบับนี้ตามลิสติงที่ 2 ดังจะนำเรื่องหลัก ๆ มากล่าวในแต่ละหัวข้อต่อไป

รับการดร็อบไฟล์เดี่ยว

แอพพลิเคชันที่ต้องการให้ผู้ใช้แดร็กแอนด์ดร็อบไฟล์เดี่ยวได้ จะต้องนำค่าที่คอมมานด์ไลน์มาใช้ ดังที่โปรแกรมนี้

ลิสติงที่ 2 (ต่อ)

```
    hPlay      : THandle;
    waPreHde   : PreHeadT;
    rMtaHde    : TMetaHeader;
    isPlceMtfF,tmrOnF,shwListF,shwSveF,clrWndF,ldeFiF : Boolean;
    vworg, vwext, wndext : TPoint;
    iNumDrpFi,iFiCnt,iTmeIntv  : Integer;

    oSveListP : PListBox;
    oldCur  : HCursor;
    mtfPath : FiNameT;
    pnt1,pnt2 : TPoint;
    aTwoPntsP : TwoPntsArrayTP;
    lftDwnF,usrSelF,drgDrpF,shwSveLstF  : Boolean;
    onTopF,mulDspF,drftRctF,useWidF : Boolean;
    hDelIco,hSaveIco,hSaveWnd : THandle;
    constructor  Init(apar : PWindowsObject; atitle : PChar);
    procedure  SetupWindow; virtual;
    procedure  Paint(paintdc : HDC;
                 var paintInfo : TPaintStruct); virtual;
    procedure  Destroy; virtual;
    procedure  wmDropFiles(var msg : TMessage);
                virtual wm_First + wm_DropFiles;
    procedure  wmCommand(var msg : TMessage);
                virtual wm_First + wm_Command;
    procedure  idCombBox(var msg : TMessage);
                virtual id_First + id_CombBox;
    procedure  wmTimer(var msg : TMessage);
                virtual wm_First + wm_Timer;
    procedure  NewFile;
    function   OpenFile : Boolean;
    procedure  LoadDspMtfInList(fwrd : Integer; disp : Boolean);
    procedure  DoLoadFi(finmeP : PChar);
    procedure  CalScale(preHde: PreHeadT);
    procedure  SetScale(adc : HDC);  virtual;
    procedure  AddFiInDir(pathB : PChar);
    procedure  ShowMainList;
    procedure  OpenNoteWnd;

    procedure  EnaDragDropMenu(frm,too : Integer; ena : Boolean);
    procedure  wmInitMenuPopup(var Msg: TMessage);
                virtual wm_First + wm_InitMenuPopup;

   (* procedure  SetOnTop;
    procedure  SaveAsMtf;
    procedure  SaveOneAsMtf(mtfFNP : PChar);
    procedure  wmLButtonDown(var msg : TMessage);
                virtual wm_First + wm_LButtonDown;
    procedure  wmLButtonUp(var msg : TMessage);
                virtual wm_First + wm_LButtonUp;
    procedure  wmMouseMove(var msg : TMessage);
                virtual wm_First + wm_MouseMove;
    procedure  wmRButtonDown(var msg : TMessage);
                virtual wm_First + wm_RButtonDown;
    procedure  DoDragDropItem(drppnt : TPoint);
    procedure  TogShowListBox(sshw : Boolean);
    procedure  wmUQstOpn(var msg : TMessage);
                virtual wm_First + wm_UQstOpn;

    procedure  RestoreItem;
    procedure  SaveListAsText;
    procedure  LoadListFromText;
    procedure  DoMulDsp; *)
                END;
   (* ------------------ WndProC  Method ----------------- *)

   procedure  SetRectFromPnts(var arct : TRect; apnt1,apnt2 : TPoint);
   begin  SetRect(arct, apnt1.x,apnt1.y, apnt2.x,apnt2.y)  end;


   FUNCTION  PntInPntsRect(pt,pt1,pt2 : TPoint) : Boolean;
   VAR  rct : TRect;
   BEGIN
     SetRect(rct, pt1.x,pt1.y, pt2.x,pt2.y);
     PntInPntsRect := PtInRect(rct, pt);
   END;


   CONST  xunit = 10;  yunit = 10;
          picwid = 420 div xunit;  picthi = 300 div yunit;
```

ได้ดำเนินการดังนี้คือ

1. ใน Init ได้ลอกค่าที่คอมมานด์-ไลน์ใน cmdLine ซึ่งเป็นตัวแปรโกลบอลของเทอร์โบปาสคาล โดยลอกให้แก่ mtfFNB เก็บไว้ ค่านี้คือชื่อไฟล์ที่ผู้ใช้แดร็กมาดร็อบ

2. ใน SetupWindow ได้ให้ DoLoadFi โหลดให้ ได้เป็นเมตาไฟล์ภาคแสดงใน hplay

3. นำออกแสดงใน Paint ซึ่งจะเป็นการแสดงเมื่อเปิดหน้าต่างครั้งแรก

รับการดร็อบไฟล์เป็นกลุ่ม

แอพพลิเคชันที่ต้องการให้ผู้ใช้แดร็กแอนด์ดร็อบไฟล์เป็นกลุ่มได้ ซึ่งสามารถดร็อบได้ทั้งในหน้าต่างหรือบนไอคอนของแอพพลิเคชันนั้น จะต้องดำเนินการดังนี้คือ

1. บ่งใช้ยูนิต Win31 และ Shell API ใน uses

2. เตรียมที่เก็บข้อมูลสตริงเพื่อเก็บชื่อไฟล์เต็ม (รวมไดรฟ์และไดเรคทอรี) ซึ่งของโปรแกรมนี้เนื่องจากต้องการนำออกแสดงด้วยจึงใช้เป็นกรอบคอมโบ ดังที่ได้ให้กำเนิด oDrpListP ใน Init โดยให้งดแสดงตัวไว้ก่อน

3. ขึ้นทะเบียนกับวินโดวส์เป็นแอพพลิเคชันที่รับการดร็อบในแบบนี้ด้วยฟังก์ชัน DragAcceptiFiles โดยให้ค่า accept เป็นจริง ดังที่ให้ดำเนินการใน SetupWindow และเมื่อจะยุติการทำงาน เช่น ใน Destroy จะต้องถอนทะเบียนด้วยฟังก์ชันนี้เช่นกัน โดยให้ค่า accept เป็นเท็จ

4. จัดกิจกรรม wmDropFiles รับเมสเสจนี้ ซึ่ง File Manager จะเป็นผู้ส่งให้เมื่อผู้ใช้แดร็กไฟล์มาดร็อบ เพื่อให้เราขอ

ลิสติงที่ 2 (ต่อ)

```
CONSTRUCTOR  WndProC.Init(apar : PWindowsObject; atitle : PChar);
BEGIN
  TWindow.Init(apar,atitle);
  attr.x := 40;  attr.y := 0;  attr.w := 500;  attr.h := 300;
  attr.menu := LoadMenu(HInstance, menuname);
  lstrcpy(mtfFNB, 'd:\book2\14mtf\wmffi\*.wmf');
  hPlay := 0;  isPlceMtfF := true;  iTmeIntv := TmeIntvK;
  waPreHde := defiso;  CalScale(waPreHde);  tmrOnF := false;

  oFiListP := New(PComboBox, Init(@self, id_CombBox,
                                  0,0,cmbWidK,200, cbs_DropDown, 79));
  with  oFiListP^.attr  do
    style := style and (not ws_Visible) and (not cbs_Sort);

  if  cmdLine[0] <> #0  then
    begin  lstrcpy(mtfFNB, cmdLine);  iNumDrpFi := 1  end
  else  iNumDrpFi := 0;  iFiCnt := 0;

  shwListF := false;  shwSveF := false;  shwSveLstF := false;
  lftDwnF := false;  drgDrpF := false;  usrSelF   := false;
  mulDspF := false;  clrWndF := false;  drftRctF  := false;
  useWidF := false;  ldeFiF  := false;  onTopF    := false;
  oSveListP := nil;
END;


PROCEDURE  WndProC.SetupWindow;
VAR   P : PChar;
BEGIN
  TWindow.SetupWindow;
  if  iNumDrpFi = 1  then  DoLoadFi(mtfFNB);

  DragAcceptFiles(HWindow, TRUE);  iNumDrpFi := 0;
  EnaDragDropMenu(2,4,false);
END;


PROCEDURE  WndProC.Destroy;
BEGIN
  if  hPlay <> 0  then  DeleteMetafile(hPlay);
  DragAcceptFiles(HWindow, FALSE);
  TWindow.Destroy;
END;


PROCEDURE  WndProC.Paint(paintDC : HDC;
                         var paintInfo : TPaintStruct);
VAR  cntdwn : Integer;  buf : array[0..9] of Char;
BEGIN
  if  clrWndF  then  begin  clrWndF := false;  Exit  end;
  if  iNumDrpFi > 1  then
    begin  cntdwn := iNumDrpFi - iFiCnt;
           WVSPrintF(buf,' %d  ', cntdwn);
           TextOut(paintDC, cmbWidK+2,0, buf, lstrlen(buf))  end;
  if  hPlay <> 0  then
    begin  SetScale(paintDC);  PlayMetaFile(paintDC, hPlay)  end;
END;


PROCEDURE  WndProC.wmDropFiles(var msg : TMessage);
VAR   cnt : Integer;  temB : array[0..127] of Char;
      hDropList : THandle;  newDrp : Integer;
BEGIN
  hDropList := msg.wParam;
  newDrp := DragQueryFile(hDropList, $FFFF, nil, 0);
  WVSPrintF(temB,'number of drop = %d', newDrp);
  MessageBox(HWindow, temB, 'Drop files', mb_OK);
  for  cnt := 0 to newDrp-1  do
    begin
    DragQueryFile(hDropList, cnt, temB, 127);
    if  StrScan(temB, '.') = nil  then  AddFiInDir(temB)
    else  oFiListP^.AddString(temB);
    end;
  DragFinish(hDropList); iNumDrpFi := oFiListP^.GetCount;
  if  IsIconic(HWindow)  then  OpenIcon(HWindow);
  EnaDragDropMenu(2,4,true);  LoadDspMtfInList(0, true);
END;
```

ชื่อไฟล์ซึ่งเก็บอยู่ในรายการชื่อไฟล์มาใช้ รายการชื่อไฟล์เป็นหน่วยความจำหน่วยหนึ่งของวินโดวส์ ซึ่ง File Manager จะเป็นผู้ให้ชื่อไฟล์เต็มตามจำนวนที่ผู้ใช้เลือก และเราจะขอมาใช้ได้ดังนี้

- File Manager จะให้ค่าแฮนเดิลของรายการชื่อไฟล์แก่ wParam ของ wm_DropFile ตามที่ได้ให้ค่าแก่ hDropList
- ขอจำนวนไฟล์ด้วยฟังก์ชัน DragQueryFile คือที่ผ่านค่า $FFFF ไปให้ นำมาเก็บไว้ใน newDrp
- ขอชื่อไฟล์ทีละไฟล์ด้วยฟังก์ชัน DragQueryFile อีกเช่นกัน (ที่อยู่ในวงรอบ for) ในครั้งนี้ความหมายของค่าที่ผ่านไปให้จะเป็นดังนี้ ขอชื่อไฟล์ใน hDropList ณ ตำแหน่งที่ cnt (จากต้นรายการ) ให้แก่ temB ซึ่งเป็นบัฟเฟอร์ขนาด 127 ไบต์
- นำชื่อไฟล์เต็มที่ขอมาได้ไปเก็บของโปรแกรมนี้ให้เก็บในกรอบคอมโบ (ส่วนตรงเงื่อนไขที่ให้ไปทำงานที่ Add FiInDir ซึ่งเป็นโปรแกรมว่างจะกล่าวในหัวข้อซ้อมมือครับ)
- เมื่อขอชื่อไฟล์หมดแล้วให้ใช้ฟังก์ชัน DragFinish เพื่อยกเลิกหน่วยความจำที่เป็นรายชื่อไฟล์เต็ม
- หากมินิไมส์เป็นไอคอน ให้เปิดหน้าต่าง
- ตามโปรแกรมนี้หลังจากที่ขอรายการไฟล์มาเก็บแล้ว ได้ให้นำออกแสดงด้วย LoadDrpMtfInList

การโหลดไฟล์

ก็คือการอ่านไฟล์ครับ เรียกชื่อให้แตกต่างกัน โปรแกรมส่วนนี้อยู่ใน DoLoadFi (มี LoadDspMtfInList เรียกใช้ด้วย) กิจกรรมนี้จะเรียกใช้ LoadMtf ได้ค่าเมตาไฟล์ภาคแสดงใน hPlay และส่วนหัว

ลิสติงที่ 2 (ต่อ)

```
PROCEDURE  WndProC.AddFiInDir(pathB : PChar);
VAR   dirInfo : TSearchRec;  bufB : array[0..79] of Char;
      P : PChar;
BEGIN

END;


PROCEDURE  WndProC.LoadDspMtfInList(fwrd : Integer; disp : Boolean);
VAR  rct : TRect;
BEGIN
  iFiCnt := iFiCnt + fwrd;
  if  iFiCnt >= iNumDrpFi  then  iFiCnt := 0;
  if  iFiCnt < 0  then  iFiCnt := iNumDrpFi;
  oFiListP^.GetString(mtfFNB , iFiCnt);  DoLoadFi(mtfFNB);
  if  disp  then  InvalidateRect(HWindow, nil, true);
  usrSelF := true;
END;


PROCEDURE  WndProC.wmTimer(var msg : TMessage);
BEGIN
  KillTimer(HWindow, TmrIdK);
  LoadDspMtfInList(1, true);
  SetTimer(HWindow, TmrIdK, iTmeIntv, nil);
END;


PROCEDURE  WndProC.idCombBox(var msg : TMessage);
BEGIN
  case  msg.lParamHi  of
    cbn_SelChange :
      begin oFiListP^.GetSelString(mtfFNB, SizeOf(mtfFnb) - 1);
            iFiCnt := oFiListP^.GetSelIndex;  usrSelF := true;
            DoLoadFi(mtfFNB); InvalidateRect(HWindow, nil, true);  end;
  end;  { case }
END;


PROCEDURE  WndProC.wmCommand(var msg : TMessage);
BEGIN
  case  msg.wParam  of
   cm_NewFi    : NewFile;
   cm_OpenFi   : OpenFile;
   cm_SveMtf   : { SaveAsMtf};
   cm_ShwLst   : ShowMainList;
   cm_ClrLst   : begin  oFiListP^.ClearList;  iNumDrpFi := 0;
                        ShowWindow(oFiListP^.HWindow, sw_Hide);
                        EnaDragDropMenu(2,4,false)  end;
   cm_ShwSve   : { begin  shwSveF := not shwSveF;
                        ShowListBox(shwSveF)  end };
   cm_ReStor   : {RestoreItem};
   cm_SveLst   : {SaveListAsText};
   cm_LdeTxt   : {LoadListFromText};
   cm_MulDsp   : {DoMulDsp};
   cm_SelFi    : usrSelF := true;
   cm_Draft    : drftRctF := not drftRctF;
   cm_UseWid   : useWidF := not useWidF;
   cm_OnTop    : {SetOnTop};
   cm_Fwrd     : begin  LoadDspMtfInList(1, true);
                        oFiListP^.SetSelIndex(iFiCnt)  end;
   cm_Bwrd     : begin  LoadDspMtfInList(-1, true);
                        oFiListP^.SetSelIndex(iFiCnt)  end;
   cm_SttTmr   : begin  SetTimer(HWindow, TmrIdK, iTmeIntv, nil);
                        tmrOnF := true  end;
   cm_StpTmr   : begin  if  not tmrOnF  then  Exit;
                        KillTimer(HWindow, TmrIdK);  tmrOnF := false  end;
   cm_Fast     : begin  if  not tmrOnF  then  Exit;
                        if  iTmeIntv >= 2000  then  dec(iTmeIntv, 1000)
                        else  if  iTmeIntv > 100  then
```

รูปที่ 5
คำสั่งที่ใช้ในการแสดงภาพ
เมื่อใช้มาตราเป็น
mm-Anisotropic

```
SetMapMode(adc, mm_Anisotropic);
SetWindowExt(adc, 3000,2000);
SetWindowOrg(adc, 1000,1000);

SetViewportExt(adc, 300,200);
SetViewportOrg(adc, 100,100);

Ellipse(adc, 1000,1000, 4000,3000);
```

เพิ่มใน waPreHde ซึ่งจะต้องนำไปคำนวณด้วย CalScale ได้ค่าสเกลใน pnt1 และ pnt2 เพื่อใช้ในการให้ค่าสเกลแก่ดีซี โดยเรียกใช้ SetScale ดังที่เห็นใน Paint

การทำความเข้าใจโปรแกรมใน CalScale และ SetScale ต้องรู้เรื่องของแม็พปิงโหมดก่อน แต่โดยที่ผมได้กล่าวถึงเรื่องนี้ไว้นานแล้ว จึงขอนำมาทบทวนก่อน

ทบทวนแม็พปิงโหมด

แม็พปิงโหมด (mapping mode) ทำให้เราสามารถเลือกมาตราในการแสดงภาพได้ มาตราหนึ่งซึ่งใช้อยู่ในเพลสเอเบิลเมตาไฟล์คือ mm_Anisotropic

มาตรานี้ทำให้สามารถแสดงภาพได้ขนาดเดียวกันโดยไม่ขึ้นกับอุปกรณ์ และส่วนมากจะอ้างเป็นนิ้ว เช่น ต้องการแสดงภาพขนาด 3x2 นิ้ว ณ ตำแหน่งคือจุดออริจิน คือจุดมุมบนซ้ายของภาพที่ 1, 1 นิ้ว ก็จะได้ภาพซึ่งขอเรียกเป็นเงา ขนาด และตำแหน่งนี้ ทั้งบนจอภาพและเครื่องพิมพ์ วิธีการ (มีรายละเอียดตามรูปที่ 5) คือ

1. เซตแม็พปิงโหมดเป็น mm_Isotropic ด้วยฟังก์ชัน SetMapping

2. กำหนดค่าอินซ์ของภาพตามแต่จะกำหนด ส่วนมากใช้ค่า 576 หรือ 1000 จะใช้เป็น 1000 เรียกเป็น LP (logical point)

3. เปลี่ยนขนาดและตำแหน่งของภาพเป็น LP จะได้ค่าเป็น 3000x2000 LP และ 1000, 1000 LP ตามลำดับ นำไปเซตขนาดและตำแหน่งของภาพด้วยฟังก์ชัน SetWindowExt และ SetWindowOrg

4. อ่านค่าอินซ์ของอุปกรณ์คือของเงา ซึ่งของจอภาพอ่านได้ด้วยฟังก์ชัน

ลิสติงที่ 2 (ต่อ)

```
                        dec(iTmeIntv, 100)  end;
    cm_Slow     : begin  if  not tmrOnF  then  Exit;
                         if  iTmeIntv >= 1000  then  inc(iTmeIntv, 1000)
                         else  inc(iTmeIntv, 100)  end;
    cm_ClrWnd   : begin  clrWndF := true;
                         InvalidateRect(HWindow, nil, true)  end;
    cm_Note     : OpenNoteWnd;
    cm_Quit     : CloseWindow;
   else
    TWindow.wmCommand(msg);
  end;
END;


PROCEDURE  WndProC.ShowMainList;
BEGIN
  shwListF := not shwListF;
  if  shwListF  then
    begin  oFiListP^.SetSelIndex(iFiCnt);
           ShowWindow(oFiListP^.HWindow, sw_Show)  end
  else   ShowWindow(oFiListP^.HWindow, sw_Hide);
END;


PROCEDURE  WndProC.OpenNoteWnd;
VAR  oWndP : PFileWindow;
BEGIN
  oWndP := New(PFileWindow, Init(@self, 'NOTE',nil));
  with  oWndP^.attr  do
    begin
    menu := LoadMenu(HInstance, editmenu);
    x := 100;  y := 10;  w := 300;  h := 200;
    style := ws_Popup or ws_OverlappedWindow or ws_Visible;
    end;
  Application^.MakeWindow(oWndP)
END;


PROCEDURE  WndProC.NewFile;
VAR  ans : Integer;  buf : FiNameT;
BEGIN
  if  hPlay <> 0  then  DeleteMetafile(hPlay);  hPlay := 0;
  InvalidateRect(HWindow, nil, true);
  lstrcpy(buf, thetitle);  SetWindowText(HWindow, buf);
END;


FUNCTION  WndProC.OpenFile : Boolean;
VAR  finme : FiNameT;
BEGIN
  NewFile; lstrcpy(finme, mtfFNB);
  if  not  GetFiNameDlg(HWindow,
                        'Window Meta file (*.WMF)¦*.wmf¦' +
                        'Simple Meta file (*.MTF)¦*.mtf¦' +
                        'All files (*.*)¦*.*¦',
                        nil, finme, true)  then  Exit;
  lstrcpy(mtfFNB, finme);   DoLoadFi(finme);
  InvalidateRect(HWindow, nil, true);
END;



PROCEDURE  WndProC.DoLoadFi(finmeP : PChar);
VAR   buf : array[0..127] of Char;
BEGIN
  if  hPlay <> 0  then  DeleteMetafile(hPlay);
  hPlay := LoadMtf(finmeP, waPreHde, rMtaHde, isPlceMtfF);
  if  not mulDspF  then  CalScale(waPreHde);

  ldeFiF := true;  lstrcpy(buf, thetitle);  lstrcat(buf, ':  ');
  if  isPlceMtfF  then  lstrcpy(buf, 'Placeable MTF:  ')
  else  lstrcpy(buf, 'Simple MTF:  ');
  lstrcat(buf, mtfFNB);  SetWindowText(HWindow, buf);
END;


PROCEDURE  WndProC.CalScale(preHde: PreHeadT);
VAR   wndDC : HDC;  dinx,diny,vinch : Integer;
BEGIN
  wndext.x := preHde[px2] - preHde[px1];
  wndext.y := preHde[py2] - preHde[py1];
  vinch := preHde[inch];
```

GetDeviceCaps เช่น ของจอภาพ VGA 640x480 จะได้ค่าอินช์เป็น 96 สมมติเป็น 100 เรียกเป็น DP (device poist)

5. หาขนาดและตำแหน่งของเงาจากค่าอินช์ทั้งสอง เช่น ความกว้างของเงาจะเป็น 3000x100*1000 ได้เป็น 300 คือ 300x200 DP และ 100, 100 DP นำไปเซตขนาดและตำแหน่งของเงาด้วยฟังก์ชัน SetViewportExt และ SetViewportOrg

6. ให้เอาต์พุตด้วยค่าภาพ ดังเช่นตามรูปที่ 4 เป็นการวาดวงรีเต็มกรอบภาพเพียงรูปเดียว จะได้เงาคือภาพบนจอภาพคือในหน้าต่างขนาด 300x200 จุดภาพ ณ ตำแหน่ง 100, 100 จุดภาพ

การใช้ฟังก์ชัน SetWindowExt กับ SetViewportExt ร่วมกันนี้และต้องใช้ร่วมกัน ผมขอเรียกเป็นการเซตค่าสเกลหรือการให้ค่าสเกลและแยกเป็นการเซตค่าสเกลของภาพกับค่าสเกลของเงาตามลำดับ และเรียกค่าที่ให้เป็นค่าสเกล

ค่าสเกลของเพลสเอเบิลเมตาไฟล์

การเซตค่าสเกลตามตัวอย่างนี้อาจเซตค่าสเกลของภาพเป็น 1000, 1000 และค่าสเกลของเงาเป็น 100, 100 จะได้ผลเช่นกัน แต่ในเพลสเอเบิลเมตาไฟล์จะเก็บคำสั่งเซตค่าสเกลของภาพไว้ในไฟล์ ให้เราเซตค่าสเกลของเงาจึงต้องดำเนินการตามตัวอย่างนี้ (หากคุณสงสัยว่าผมทราบได้อย่างไร เรื่องยาวครับ ผมใช้เวลาเป็นเดือน จนต้องเขียนโปรแกรมอ่านค่าในไฟล์ .WMF ออกแสดงจึงทราบ)

ที่เป็นเช่นนี้ผมเข้าใจว่า หากเราต้องการให้แสดงภาพตามขนาดและตำแหน่งเป็นนิ้วตามที่ผู้วาดต้องการ ให้

ลิสติงที่ 2 (ต่อ)

```
  wndDC := GetDC(HWindow);
  dinx := GetDeviceCaps(wndDC, LogPixelsx);
  diny := GetDeviceCaps(wndDC, LogPixelsy);

  vwext.x := MulDiv(wndext.x, dinx, vinch);
  vwext.y := MulDiv(wndext.y, diny, vinch);

  SetMapMode(wndDC, mm_Anisotropic);
  SetWindowExt(wndDC, wndext.x, wndext.y);
  SetViewportExt(wndDC, vwext.x, vwext.y);
  vworg.x := preHde[px1];  vworg.y := preHde[py1];
  LPtoDP(wndDC, vworg, 1);
  ReleaseDC(HWindow, wndDC);

  pnt1 := vworg;  pnt2.x := pnt1.x + vwext.x;
  pnt2.y := pnt1.y + vwext.y;
END;


PROCEDURE  WndProC.SetScale(adc : HDC);
VAR  vWid,vHi : Integer;
BEGIN
  SetMapMode(adc, mm_Anisotropic);
  if  useWidF  then
    begin  vWid := pnt2.x-pnt1.x;
           vHi := MulDiv(wndext.y, vWid, wndext.x);
           SetViewportExt(adc, vWid, vHi)  end
  else  SetViewportExt(adc, pnt2.x-pnt1.x, pnt2.y-pnt1.y);
  SetViewportOrg(adc, pnt1.x, pnt1.y);
END;

(* ---------------------- Menu routine ---------------------- *)

PROCEDURE  WndProC.EnaDragDropMenu(frm,too : Integer; ena : Boolean);
VAR  pstn,act,icm : Word;  hFi : HMenu;
BEGIN
  act := mf_ByPosition;
  if  ena  then  act := act or mf_Enabled  else  act := act or mf_Grayed;
  for  pstn := frm to too  do
    EnableMenuItem(attr.menu, pstn, act);
  DrawMenuBar(HWindow);
END;


PROCEDURE  WndProC.wmInitMenuPopup(var Msg: TMessage);
VAR   hFi,hMisc : HMenu;  icm,act : Integer;
      newItmB : array[0..79] of Char;
BEGIN
  hFi := GetSubMenu(attr.menu, 0);
  if  msg.wParam = hFi  then
    begin
    if  ldeFiF  then
      EnableMenuItem(hFi, cm_SveMtf, mf_ByCommand or mf_Enabled)
    else  EnableMenuItem(hFi, cm_SveMtf, mf_ByCommand or mf_Grayed);

    if  shwListF  then  lstrcpy(newItmb, 'Hide &Files List')
    else  lstrcpy(newItmb, 'Show &Files List');
    ModifyMenu(hFi, cm_ShwLst, mf_ByCommand, cm_ShwLst, newItmB);

    if  shwSveF  then  lstrcpy(newItmb, 'Hide Sa&ve List')
    else  lstrcpy(newItmb, 'Show Sa&ve List');
    ModifyMenu(hFi, cm_ShwSve, mf_ByCommand, cm_ShwSve, newItmB);


    if  iNumDrpFi > 0  then
      EnableMenuItem(hFi, cm_ClrLst, mf_ByCommand or mf_Enabled)
    else  EnableMenuItem(hFi, cm_ClrLst, mf_ByCommand or mf_Grayed);

    if  oSveListP = nil  then  Exit;
    if  oSveListP^.GetCount > 0  then
      EnableMenuItem(hFi, cm_ReStor, mf_ByCommand or mf_Enabled)
    else  EnableMenuItem(hFi, cm_ReStor, mf_ByCommand or mf_Grayed);
    Exit
    end;

  hMisc := GetSubMenu(attr.menu, 1);
  if  msg.wParam <> hMisc then  Exit;
  if  iNumDrpFi = 0  then  act := mf_Grayed  else  act := mf_Enabled;
  EnableMenuItem(hMisc, cm_SveLst, mf_ByCommand or act);
  EnableMenuItem(hMisc, cm_MulDsp, mf_ByCommand or act);
```

คำนวณค่าสเกลของเงาตามที่กล่าวมา แต่ถ้าเราต้องการแสดงภาพคือเงา ณ ตำแหน่งและขนาดใหม่ก็สามารถให้ค่าได้โดยตรง เช่น อาจให้ค่าสเกลของเงาเป็น 200x100 DP ดังที่ผมนำมาใช้ให้ผู้ใช้แสดงภาพด้วยการตีกรอบ ในโปรแกรมนี้จึงให้ดำเนินการดังนี้

การคำนวณค่าสเกล

การคำนวณค่าสเกลอยู่ในกิจกรรม CalScale คือ ทุกครั้งที่ DoLoadFi ให้ LoadMtf อ่านไฟล์ให้จะได้ส่วนหัวเพิ่มใน waPreHde ซึ่งมีค่ากรอบของภาพและค่าอินช์ของภาพ ส่งให้ CalScale ดำเนินการ

โดยที่ผมใช้ CalScale ในโปรแกรม WnfPaint ด้วย จึงให้ดำเนินการดังนี้คือ

1. คำนวณค่าความกว้างและความสูงของภาพให้แก่ wndext และให้ค่านิ้วของภาพแก่ vinch

2. อ่านค่านิ้วของจอภาพคือของเงาให้แก่ dinx และ diny

3. คำนวณค่าความกว้างและความสูงของเงาให้แก่ vwext

4. ให้ค่าสเกลแก่ wndDC เพื่อที่จะใช้ฟังก์ชัน LPtoDP เปลี่ยนค่าจุดมุมของภาพใน vworg เป็นจุดมุมของเงา

5. สำหรับโปรแกรมที่เป็นพิเศษคือ เมื่อให้ผู้ใช้ตีกรอบ จะได้ค่ากรอบ (จุดมุมทั้งสอง) ใน pnt1 และ pnt2 เพื่อให้เหมือนกันจึงนำค่าใน vworg และ vwext มาให้แก่ pnt1 และ pnt2

การให้ค่าสเกลแก่ดีซี

เมื่อจะแสดงภาพจะต้องให้ค่าสเกลแก่ดีซีด้วยการเรียกใช้ SetScale ดังที่ใน Paint ให้แก่ paintDC โดยที่การให้ค่าสเกลใน SetScale จะให้ค่าสเกลของเงา

ฉัสติงที่ 2 (ต่อ)

```
  if  mulDspF  then  lstrcpy(newItmb, '&Single display')
  else  lstrcpy(newItmb, '&Multiple display');
  ModifyMenu(hMisc, cm_MulDsp, mf_ByCommand, cm_MulDsp, newItmB);

  if  drftRctF  then
    CheckMenuItem(hMisc, cm_Draft, mf_ByCommand or mf_Checked)
  else  CheckMenuItem(hMisc, cm_Draft, mf_ByCommand or mf_UnChecked);

  if  useWidF  then
    CheckMenuItem(hMisc, cm_UseWid, mf_ByCommand or mf_Checked)
  else  CheckMenuItem(hMisc, cm_UseWid, mf_ByCommand or mf_UnChecked);

  if  onTopF  then
    CheckMenuItem(hMisc, cm_OnTop, mf_ByCommand or mf_Checked)
  else  CheckMenuItem(hMisc, cm_OnTop, mf_ByCommand or mf_UnChecked)
END;


(* ------------------  TheAppC Class  ---------------- *)

TYPE
  TheAppC = OBJECT(TApplication)
    procedure  InitMainWindow; virtual;
            END;


PROCEDURE  TheAppC.InitMainWindow;
BEGIN
  MainWindow := New(WndProCP, Init(nil,thetitle));
END;

(* --------------------  MAIN PROGRAM  ----------------- *)

VAR   app :  TheAppC;

BEGIN
  app.Init(appname);
  app.Run;
  app.Done;
END.
```

เพียงอย่างเดียว (ตามที่กล่าวมา) และทำให้เกิดภาพได้ 3 ลักษณะคือ

1. ตรง else ค่าที่ให้อาจเป็นค่าที่คำนวณจากของภาพใน CalScale จะทำให้ได้ภาพตามขนาดและตำแหน่งที่ผู้วาดต้องการ ดังภาพเฮลิคอปเตอร์ลำกลางในรูปที่ 2

2. ตรง else เช่นกัน ซึ่งอาจเป็นค่าที่เกิดจากการดึกรอบของผู้ใช้ เมื่อนำมาใช้ตรง ๆ จึงอาจได้ภาพที่ไม่สมส่วนดังภาพเฮลิคอปเตอร์ลำบน

3. ตรง if คือเมื่อผู้ใช้เลือกรายการ Misc/Use Width to Scale แล้วดึกรอบจะใช้ค่าสเกลในทางกว้างกับทางสูงด้วย ทำให้ได้ภาพที่สมส่วน แต่อาจเล็กหรือใหญ่กว่าภาพเดิม ดังภาพเฮลิคอปเตอร์ลำล่าง

โปรแกรมที่เหลือ

โปรแกรมที่เหลือผมคิดว่าอ่านได้ไม่ยาก จึงขอนำแค่หน้าที่ของกิจกรรมบางกิจกรรมมากล่าวถึง คือ

LoadDspMtfInList อ่านไฟล์ ณ ตำแหน่ง iFiCnt ในกรอบรายการไฟล์ ซึ่งอาจให้เพิ่มหรือลดค่า iFiCnt ก่อนตามค่าที่ให้แก่ fwrd เป็นการเพิ่มลดแบบวงรอบ ซึ่งอาจให้อ่านอย่างเดียวหรือให้แสดงด้วยโดยให้ disp เป็นจริง

idCombBox เมื่อผู้ใช้เลือกไฟล์ในกรอบรายการจะมาทำงานที่กิจกรรมนี้ ซึ่งจะอ่านไฟล์ออกแสดง

OpenNoteWnd จะเปิดหน้าต่างของสกุล TFileWindow เป็นหน้าต่างลอยเพื่อให้ผู้ใช้บันทึกข้อความที่ต้องการ

EnaDragDropMenu จะเอนเอเบิลรายการที่เมนูหลักตั้งแต่ frm ถึง too เมื่อให้ ena เป็นจริง และดิสเอเบิลเมื่อเป็นเท็จ

wmInitMenuPopup เมื่อผู้ใช้เลือกรายการที่เมนูหลักเช่น File เพื่อเปิดเมนูรอง วินโดวส์จะส่งเมสเสจและมาทำงานที่กิจกรรมนี้ ซึ่งจะนำแฟล็ก (ฟิลด์ข้อมูลที่เป็นบูลีน) มาตรวจเพื่อ

- เอนเอเบิลและดิสเอเบิลรายการ ซึ่งจะดิสเอเบิลเมื่อรายการนั้นไม่อยู่ในสถานะที่จะให้ผู้ใช้เลือก

- ให้เครื่องหมายเช็คหน้ารายการเพื่อแสดงว่าเป็นรายการที่มีสถานะค้าง เช่น Misc/With Rectangle และยกเลิกเครื่องหมายนี้เมื่อเลือกซ้ำ

- เปลี่ยนชื่อรายการเมื่อรายการนั้นมีสถานะกลับกลอก (toggle) เช่น จาก Show เป็น Hide และกลับกัน เป็นต้น

รีซอร์ส

ขอให้สร้างรีซอร์สเก็บไว้ในไฟล์ readmtf.res ดังนี้

1. เมนูรีซอร์สตามรูปแบบในรูปที่ 6 ให้ชื่อว่า readmtf_menu

2. ไอคอนรีซอร์สมี 2 รีซอร์สคือ

- ทำเป็นรูปถังผง ให้ชื่อว่า delico_ico

- ทำเป็นรูปกล่องเก็บของ ให้ชื่อว่า sveico_ico

3. เคอร์เซอร์รีซอร์ส ทำเป็นรูปไฟล์คือบัตรหลายไฟล์ ให้ชื่อว่า drgdrp_cur

ข้อมือ

มีเรื่องให้ข้อมือร่วมสนุกหลายเรื่องครับ คือ

```
READMTF_MENU MENU
BEGIN
        POPUP "&File"
        BEGIN
                MENUITEM "&New", 101
                MENUITEM "&Load", 102
                MENUITEM "&Save As (MTF)", 103
                MENUITEM SEPARATOR
                MENUITEM "Show &Files List", 104
                MENUITEM "&Clear List ", 105
                MENUITEM SEPARATOR
                MENUITEM "Show &Save List", 106
                MENUITEM "&Restore", 107
                MENUITEM SEPARATOR
                MENUITEM "&Note", 108
                MENUITEM SEPARATOR
                MENUITEM "&Exit", 109
        END

        POPUP "&Misc"
        BEGIN
                MENUITEM "&Save List As Text", 201
                MENUITEM "&Load From Text", 202
                MENUITEM SEPARATOR
                MENUITEM "&Multiple Display", 203
                MENUITEM "&Select File", 204
                MENUITEM SEPARATOR
                MENUITEM "&With Rectangle", 205
                MENUITEM "&Use Width to Scale", 206
                MENUITEM SEPARATOR
                MENUITEM "Allways on &Top", 209
        END

        MENUITEM "For&ward", 300
        MENUITEM "&Backward", 400
        POPUP "&Timer"
        BEGIN
                MENUITEM "&Start Timer", 501
                MENUITEM "S&top Timer", 502
                MENUITEM SEPARATOR
                MENUITEM "&Fast", 503
                MENUITEM "S&low", 504
        END

        MENUITEM "&Clear", 600
END
```

รูปที่ 6 รูปแบบของเมนูรีซอร์สของโปรแกรม ReadMtf

1. ในการแดร็กไฟล์เป็นกลุ่มสามารถที่จะแดร็กไดเรคทอรีไปได้ด้วย ซึ่งจะต้องเขียนโปรแกรมรองรับ ตามที่ได้เตรียมกิจกรรม AddFilnDir เป็นโปรแกรมเปล่าไว้ เพื่อจะเขียนโปรแกรมอ่านชื่อไฟล์จากไดเรคทอรีมาเก็บในกรอบรายการไฟล์ (ใช้ FindFirst กับ FindNext ดูตัวอย่างได้ในเรื่อง "โอโอพีกับรีเคอร์ชัน" ไมโครฯ 111)

2. การแดร็กแอนด์ดร็อบในหน้าต่างที่ให้แดร็กภาพไปปล่อยในถังผงหรือกล่องเก็บซึ่งยังไม่มีโปรแกรม ขอให้นึกถึงเรื่องถังผงอย่างเดียวก่อน เรื่องนี้ถ้าไม่คิดก็จะนึกว่าง่าย แต่ถ้าคิดจะเห็นว่าไม่ใช่ แต่พอรู้เรื่องแล้วจะเห็นว่าง่ายมาก

ต้องลองคิดเขียนโปรแกรมครับ คือคิดว่าจะให้ผู้ใช้ทำอะไรและเราจะเขียนโปรแกรมอย่างไร โดยใช้คำสั่งสมมติสำหรับการเขียนโปรแกรมบนวินโดวส์คือคำที่คิดว่าน่าจะมีในวินโดวส์ เสร็จแล้วนำไปค้นคำเช่นใน Help หากเคยผ่านตามาบ้างก็ให้พยายามนึก หรือนำชื่อเรื่อง ชื่อที่เกี่ยวพัน ไปค้น

บางเรื่องก็ใช้เวลาไม่นาน บางเรื่องก็หลายวัน (ไม่ได้ค้นทั้งวัน) บางเรื่องก็ค้นไม่พบ ต้องเปลี่ยนวิธีการ แต่ก็เป็นวิธีหนึ่งที่จะใช้ในการศึกษาการเขียนโปรแกรมบนวินโดวส์ครับ ดังที่ผมใช้อยู่

เมื่อค้นแล้วขอให้ค้นต่ออีกเรื่องครับคือ ในการแดร็กดร็อบไฟล์เป็นกลุ่ม หากทำให้หน้าต่างของโปรแกรมนี้อยู่บนหน้าต่างของ File Manager ตลอดเวลาจะทำให้แดร็กดร็อบง่ายขึ้น ดังที่ผมได้เตรียมรายการ Always on Topไว้ใน Misc และเตรียมชื่อกิจกรรม SetOnTop ไว้แล้ว

หนังสือใหม่

หนังสือ "การเขียนโปรแกรมบนวินโดวส์ ด้วยเทอร์โบปาสคาล" เล่ม 1 ของผมออกแล้ว (ตั้งแต่ ก.ย.37)

ไม่โฆษณาก็เหมือนโฆษณาครับ เพราะระยะหลังนี้ผมอืดอัดใจพอสมควร เนื่องจากเรื่องที่เขียนเรียกได้ว่าเป็นโปรแกรมใช้งาน จึงมีทั้งเรื่องใหม่ที่เป็นเรื่องเฉพาะเป็นหัวข้อในการเขียนครั้งนั้น กับเรื่องที่เป็นพื้นฐานในการเขียนโปรแกรมบนวินโดวส์ เป็นเรื่องเก่าที่เคยกล่าวถึงไปแล้ว หากไม่นำมากล่าวก็เกรงว่าผู้ที่พึ่งเริ่มอ่านจะบ่น เมื่อนำมากล่าวซ้ำก็เกรงว่าผู้ที่ทราบแล้วจะรำคาญ

และด้วยเหตุนี้ด้วยที่ทำให้ผมตัดสินใจแบ่งหนังสือออกเป็น 2 เล่ม เล่มแรกนี้มีแต่เรื่องที่เป็นพื้นฐาน ได้แก่ หน้าต่าง เมนู คอนโทรล ไดอะล็อกซ์ และรีซอร์ส

ในการศึกษาการเขียนโปรแกรม

ผมคิดว่าควรมีหนังสืออย่างน้อย 1 เล่ม ไม่จำเป็นต้องเป็นของผม แต่เมื่อมีหนังสือเล่มนี้ในท้องตลาดทำให้ผมมั่นใจว่าจะหาอ่านกันได้ จะได้เขียนเรื่องเดินหน้าต่อไปได้ครับ และต่อไปผมคงต้องอ้างถึงบ้างสำหรับผู้ที่มีเพื่อให้ค้นได้ง่าย

แต่ถ้าไม่มีผมหมายถึงไม่มีหนังสือของใครเลย ก็เหมือนผมเมื่อเริ่มแรกมีหนังสือภาษาซีอยู่เพียงเล่มเดียว ต้องอาศัยดูตัวอย่างในเทอร์โบปาสคาล เป็นการศึกษาจากโปรแกรมโดยตรง

ซอฟต์เมล์

ผมได้มอบโปรแกรมรวมทั้งไฟล์ .EXE ที่ใช้ในฉบับนี้และฉบับหน้า ดังนี้ครับ

1. ยูนิต UBkUtils ทั้งไฟล์ TPU และ TPW (เป็นไฟล์ในหนังสือ ใช้สำหรับการให้กำเนิดไฟล์ไดอะล็อกบ็อกซ์ของวินโดวส์)

2. ไฟล์สำหรับผนวกคือ ldesvw-mf.icl และ icownd.icl

3. ไฟล์รีซอร์สคือ readmtf.res

4. ไฟล์ .pas คือ readmtf.pas

และผมได้ให้ไฟล์ readmtf.exe มาด้วยสำหรับท่านที่ต้องการทดลองรันโปรแกรม

ท้ายตอน

ในฉบับนี้ผมได้นำเรื่องของการเขียนโปรแกรมบนวินโดวส์มากล่าวถึงหลายเรื่องด้วยกันคือ

พื้นฐาน ได้แก่

1. การสร้างหน้าต่างหลัก (สกุล WndProC) และการให้กำเนิด (ใน The AppC.InitMainWindow) ซึ่งเป็นพื้นของพื้น คงไม่กล่าวถึงอีกครับ

2. การรับเมสเสจจากเมนู (ใน wmCommand)

3. การดำเนินการกับเมนู (ใน EnaDragDropMenu และ wmInitMenu-Popup)

4. การใช้ไทเมอร์ (wmCommand และ wmTimer)

5. การใช้งานกรอบคอมโบ (ในที่หลายแห่งด้วยกัน) ได้แก่ การให้กำเนิด การเพิ่มรายการ อ่านรายการ และการให้แสดงตัวและงดแสดงตัว

6. การให้กำเนิดหน้าต่างลูก (คือ Note)

เรื่องเฉพาะ ได้แก่

1. การอ่านและเขียนกับไฟล์เพลสเอเบิลเมตาไฟล์ ตลอดจนการนำออกแสดง

2. การแดร็กแอนด์ดร็อบทั้ง 2 วิธี

ฉบับหน้าจะเป็นเรื่องของการแดร็กแอนด์ดร็อบภาพในหน้าต่าง การแปลงเพลสเอเบิลเมตาไฟล์เป็นเมตาไฟล์ธรรมดา และการทำเป็นรีซอร์สครับ

หนังสือดี ซีเอ็ด

การพัฒนาระบบฐานข้อมูลด้วยภาษาซี

■ โดย ออล สตีเวนส์
เรียบเรียงโดย อนุชิต สุมชิวิทย์/วีร์ ตั้งมั่นภักดีพงษ์
ราคาเล่มละ 150 บาท

หนังสือเล่มนี้ประกอบด้วยพื้นฐานและการออกแบบข้อมูลเบื้องต้น วิธีการสร้างระบบจัดการฐานข้อมูลอย่างง่ายด้วยภาษาซี พร้อมโปรแกรมที่สามารถนำไปใช้งานได้จริง ผู้อ่านจึงควรมีความสามารถในการเขียนโปรแกรมด้วยภาษาซีอยู่บ้างแล้ว นอกจากนี้จุดประสงค์ของหนังสือต้องการให้ผู้อ่านสามารถสร้าง tool เพื่อลดเวลาในการโปรแกรมลงได้อีกด้วย หนังสือเล่มนี้จึงเหมาะสำหรับผู้ที่สนใจเรื่องฐานข้อมูล และผู้ที่รักการเขียนโปรแกรมทั่วไป

หาซื้อได้ตามร้านหนังสือทั่วไป
หรือสั่งซื้อโดยตรงได้ที่

บริษัท ซีเอ็ดยูเคชั่น จำกัด
800/43-45 ซอยตระกูลสุข ถนนอโศก-ดินแดง ห้วยขวาง กรุงเทพฯ 10400 โทร. 248-6280-9 (เวลาทำการ จันทร์-ศุกร์ 8.10-17.30 น.) โดยส่งธนาณัติสั่งจ่าย ปท.ดินแดง ในนามบริษัทฯ หรือเช็คส่วนตัวของธนาคารในเขตกรุงเทพฯ พร้อมค่าจัดส่ง
ค่าจัดส่ง สั่งซื้อไม่เกิน 100 บาท ค่าจัดส่ง 8 บาท สั่งซื้อเกิน 100 บาท ค่าจัดส่ง 8%
พิเศษ สำหรับสมาชิกวารสารของซีเอ็ด สั่งซื้อโดยตรงที่บริษัทฯ ได้ในราคาลด 15% เพิ่มค่าลงทะเบียน 4 บาท ค่าจัดส่งฟรี โดยระบุหมายเลขสมาชิกด้วย